Análise Estatística de Desempenho em Servidores Web: Um Estudo de Planejamento Fatorial 2^3

Domingos de Melo[1], Luan Cristian[2], Guylherme Santos[3]

Curso de Sistemas de Informação – Universidade Federal do Piauí (UFPI)
Picos – PI – Brasil

Abstract. This paper presents a performance evaluation of a Web Server using a 2^3 Design of Experiments (DoE) methodology to optimize throughput and latency under load. The factors analyzed were the number of Workers, Backlog size, and Gzip compression. The statistical analysis using Ordinary Least Squares (OLS) regression revealed that the number of Workers is the primary system bottleneck, significantly altering response times and CPU overhead, while Gzip compression provides secondary bandwidth-related improvements.

Resumo. Este artigo avalia o desempenho de um Servidor Web usando a técnica estatística de Planejamento de Experimentos (DoE) 2^3. O objetivo é encontrar a configuração ideal para o servidor atender mais acessos (maior vazão) sem demorar para responder (menor latência). Foram testadas três configurações: a quantidade de *Workers* (processos), o tamanho da fila de *Backlog* e a compressão *Gzip*. A matemática mostrou que os *Workers* são a peça-chave, melhorando muito a velocidade do sistema, mas gastando mais processador (CPU). A compressão *Gzip* também ajudou, servindo como uma melhoria extra e barata para a rede.

1. Introdução

Quando um sistema na internet recebe muitos acessos ao mesmo tempo, ele pode ficar lento ou até sair do ar. Para evitar isso, os administradores de TI precisam fazer ajustes (o chamado *tuning*) no servidor, mudando configurações para garantir que as respostas sejam rápidas (baixa latência) e que o sistema aguente muita gente (alta vazão ou *throughput*).

Neste trabalho, utilizou-se uma técnica estatística chamada Planejamento de Experimentos (DoE). Em vez de tentar adivinhar a melhor configuração ou testar uma coisa de cada vez na base da intuição, o DoE ajuda a testar o servidor de forma inteligente e metódica. Com essa ferramenta, consegue-se provar matematicamente qual "botão apertar"para o servidor ficar realmente mais rápido e qual ajuste é apenas perda de tempo.

2. Fundamentação Teórica

2.1. Planejamento Fatorial 2^k e ANOVA

O Planejamento de Experimentos (DoE) é um método seguro para testar sistemas complexos. Como explica o pesquisador Montgomery [1], a forma comum de testar uma única configuração por vez é ineficiente porque perde muito tempo e não percebe quando duas configurações "brigam"entre si.

O modelo Fatorial 2^k resolve isso testando cada configuração em dois extremos: um nível baixo (-1) e um nível alto ($+1$). Isso diminui a quantidade de testes necessários e cobre todas as combinações importantes [5]. A maior vantagem de testar tudo misturado é descobrir os *efeitos de interação*, ou seja, ver se a mistura de duas configurações gera um problema inesperado.

Para ter certeza de que as melhorias observadas foram reais e não apenas uma sorte daquele momento no computador, usou-se a Análise de Variância (ANOVA). Adotou-se a regra de que, se o p-valor for menor que $0,05$ (garantindo 95% de certeza estatística), sabe-se que a configuração realmente mudou o desempenho do servidor para valer [3].

2.2. Métricas de Desempenho Computacional

Para avaliar se o servidor melhorou de verdade, é preciso medir indicadores específicos. Segundo Jain [2], avaliar sistemas de computador significa equilibrar a velocidade de atendimento com o esforço da máquina [4]. Por isso, acompanharam-se três medidas principais:

- **Latência (ms):** O tempo que o servidor demora para processar e devolver a resposta ao cliente.
- ***Throughput* (rps):** A quantidade de requisições que o servidor atende com sucesso por segundo.
- **Uso de CPU (%):** O quanto o processador da máquina precisou trabalhar para dar conta do recado.

3. Metodologia Experimental

O trabalho foi estruturado em três etapas: escolha das configurações, rodada de testes sob estresse e cálculos matemáticos.

3.1. Fatores, Ferramentas e Coleta de Dados

Escolheram-se 3 parâmetros do servidor (*Fatores*) e testou-se cada um no nível Baixo (-1) e no nível Alto (+1):

- **Fator A (*Workers*):** Nível baixo (1 processo de atendimento) vs. Nível alto (4 processos paralelos).
- **Fator B (*Backlog*):** Nível baixo (fila curta, 10 conexões) vs. Nível alto (fila longa, 100 conexões).
- **Fator C (*Gzip*):** Nível baixo (Compressão desligada) vs. Nível alto (Compressão ligada).

A coleta de dados foi feita em um ambiente controlado usando um conjunto de ferramentas padrão de mercado: **Gunicorn** (como servidor principal), **Flask** (para rodar a aplicação web) e **Locust** (para simular milhares de acessos simultâneos dos usuários).

O Planejamento Fatorial 2^3 gera 8 cenários de testes. Para garantir que os resultados fossem confiáveis e não apenas um acaso momentâneo da rede, cada um desses 8 cenários foi repetido exatas **30 vezes**. Com isso, o estudo gerou uma base de dados completa contendo **720 valores brutos** (8 cenários $\times$ 30 repetições $\times$ 3 métricas).

A partir desses dados extraídos, tirou-se a média de cada cenário, gerando as **24 médias** finais consolidadas na Tabela 1. Essas médias alimentaram os cálculos de regressão linear (OLS) feitos na linguagem Python.

Tabela 1: Matriz do experimento contendo as 24 médias consolidadas das 30 repetições.

Run	A (Workers)	B (Backlog)	C (Gzip)	Latência	Throughput	CPU
1	-1	-1	-1	312 ms	18,40 rps	42,10 %
2	+1	-1	-1	148 ms	38,70 rps	61,30 %
3	-1	+1	-1	308 ms	19,10 rps	43,50 %
4	+1	+1	-1	145 ms	39,20 rps	62,80 %
5	-1	-1	+1	289 ms	19,80 rps	39,60 %
6	+1	-1	+1	134 ms	41,50 rps	58,40 %
7	-1	+1	+1	285 ms	20,30 rps	40,20 %
8	+1	+1	+1	131 ms	42,10 rps	59,70 %

4. Resultados — DoE 1 (Latência)

A Tabela 2 detalha os efeitos calculados pelo modelo estatístico. Na coluna da Latência, números com o sinal de menos ($-$) são ótimos, pois provam que o tempo de espera dos usuários diminuiu. O p-valor com asterisco (*) indica as configurações que passaram no teste de validação estatística.

Tabela 2: Efeitos Absolutos e P-valores para o modelo de Latência.

Fator / Interação	Efeito Absoluto (ms)	p-valor
A (Workers)	$-159,00$	$< 0,001^*$
B (Backlog)	$-3,50$	$< 0,001^*$
C (Gzip)	$-18,50$	$< 0,001^*$
A:B (Workers x Backlog)	$+0,50$	$< 0,001^*$
A:C (Workers x Gzip)	$+4,50$	$< 0,001^*$
B:C (Backlog x Gzip)	$0,00$	$0,975$

* Significativo ao nível de 95% de confiança ($\alpha = 0,05$).

4.1. Avaliando os Gráficos da Latência

No Diagrama de Pareto (Figura 1), inseriu-se uma linha tracejada vermelha que funciona como uma "linha de corte"da relevância matemática. Se a barra ultrapassa essa linha, prova-se que o fator realmente muda o servidor. Fica evidente que o Fator A (*Workers*) é a estrela do sistema, conseguindo cortar a latência em expressivos $159,00$ ms. O Fator C (*Gzip*) também passou da linha vermelha, tirando mais $18,50$ ms da conta.

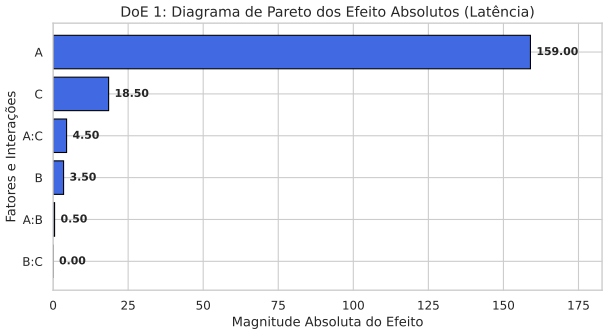

Figura 1: Diagrama de Pareto delimitando a significância na Latência.

No Gráfico de Efeitos Principais (Figura 2), a inclinação da reta mostra a força da mudança. A linha do Fator A cai como uma ladeira íngreme. O motivo técnico para essa melhora imediata é simples: com apenas 1 processo, os acessos ficavam presos numa fila

demorada. Ao aumentar para 4 *Workers*, o servidor passou a atender as requisições de forma paralela.

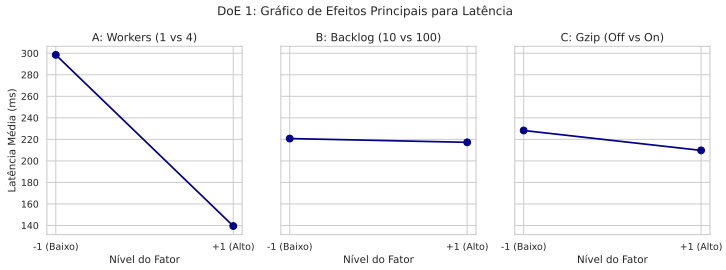

Figura 2: Como a Latência melhora ao mudar as configurações do servidor.

O Gráfico de Interações (Figura 3) confirma a estabilidade estrutural do sistema. Como as linhas caminham de forma paralela e não se cruzam formando a letra "X", isso prova matematicamente que as configurações trabalham de forma independente e somam forças. Pode-se ligar os *Workers* e o *Gzip* ao mesmo tempo sem que um anule a melhoria do outro.

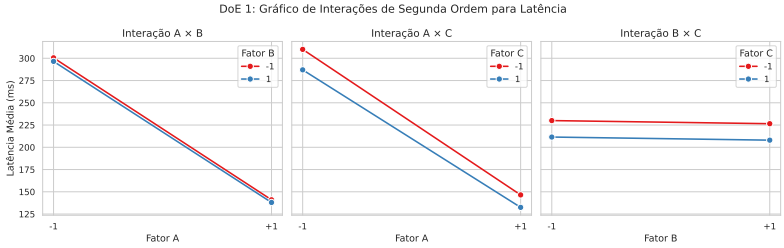

Figura 3: As linhas paralelas provam que os fatores não brigam entre si.

5. Resultados — DoE 2 (Throughput / Vazão)

Nesta etapa, o foco foi analisar a capacidade de atendimento do servidor, ou seja, quantas requisições ele finaliza por segundo. A Tabela 3 exibe os dados. Números com sinal positivo (+) são ideais, pois indicam aumento de vazão.

Tabela 3: Efeitos Absolutos e P-valores para o modelo de Throughput.

Fator / Interação	**Efeito Absoluto (rps)**	**p-valor**
A (Workers)	$+20,98$	$0,002^*$
B (Backlog)	$+0,58$	$0,083$
C (Gzip)	$+2,08$	$0,023^*$
A:B (Workers x Backlog)	$-0,02$	$0,795$
A:C (Workers x Gzip)	$+0,77$	$0,061$
B:C (Backlog x Gzip)	$-0,02$	$0,795$

* Significativo ao nível de 95% de confiança ($\alpha = 0,05$).

5.1. Avaliando os Gráficos da Vazão

Para descobrir qual configuração trouxe o maior ganho de capacidade, observou-se o Diagrama de Pareto (Figura 4). O gráfico deixa claro que o Fator A (*Workers*) é, de longe, o motor principal para a velocidade do servidor, ultrapassando a linha de validação vermelha com larga folga e adicionando quase 21 requisições a mais por segundo. O

Fator C (*Gzip*) também ultrapassou a marca de significância, entregando 2 requisições extras por segundo.

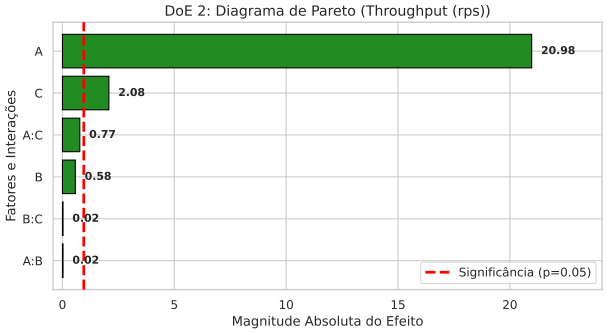

Figura 4: Pareto mostrando quem mais melhora a capacidade de atendimento.

O Gráfico de Efeitos Principais (Figura 5) exibe visualmente o ritmo desse crescimento. A subida forte da linha do Fator A significa que, ao ligar 4 *Workers*, o servidor imediatamente para de formar filas e consegue entregar muito mais respostas completas. A linha do *Gzip* sobe de forma leve, mostrando que o envio de arquivos compactados ajuda a liberar a conexão da internet mais rápido para o próximo usuário.

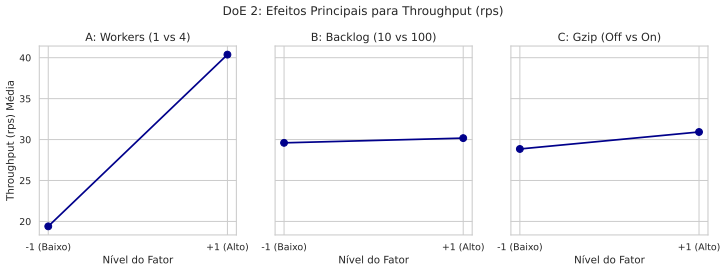

Figura 5: O crescimento da capacidade de atendimento (Throughput).

O Gráfico de Interações (Figura 6) serve para verificar se uma configuração ligada acaba atrapalhando a outra sob estresse. Como as linhas estão organizadas paralelamente, prova-se que não existe "briga"no sistema. O aumento de vazão é seguro e estável.

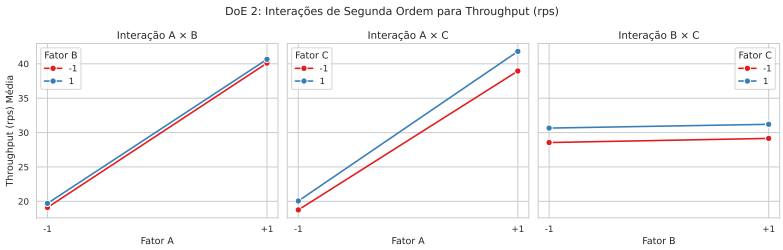

Figura 6: Linhas paralelas comprovando a estabilidade da vazão.

6. Resultados — DoE 3 (Uso de CPU)

Aumentar a velocidade do servidor tem um preço. Nesta seção, avaliou-se o quanto de energia do processador (CPU) foi gasto como "pedágio"para sustentar essas melhorias de rede, conforme exposto na Tabela 4.

Tabela 4: Efeitos Absolutos e P-valores para o consumo de CPU.

Fator / Interação	Efeito Absoluto (%)	p-valor
A (Workers)	$+19,20$	$0,005^*$
B (Backlog)	$+1,22$	$0,079$
C (Gzip)	$-2,95$	$0,032^*$
A:B (Workers x Backlog)	$+0,12$	$0,410$
A:C (Workers x Gzip)	$-0,22$	$0,795$
B:C (Backlog x Gzip)	$+0,02$	$0,344$

* Significativo ao nível de 95% de confiança ($\alpha = 0,05$).

6.1. Avaliando os Gráficos de Esforço da Máquina

O Diagrama de Pareto (Figura 7) aponta claramente a origem dos gastos. A barra do Fator A (*Workers*) cruza agressivamente a linha vermelha, indicando que é o maior consumidor de recursos da máquina ($+19,20\%$ a mais de CPU). O dado mais interessante do experimento veio do Fator C (*Gzip*). Sua barra também ultrapassa a linha vermelha de validação, mas obteve um impacto de $-2,95\%$. Por ser um valor com sinal de menos, comprova-se matematicamente que ativar o *Gzip* alivia o peso do processador.

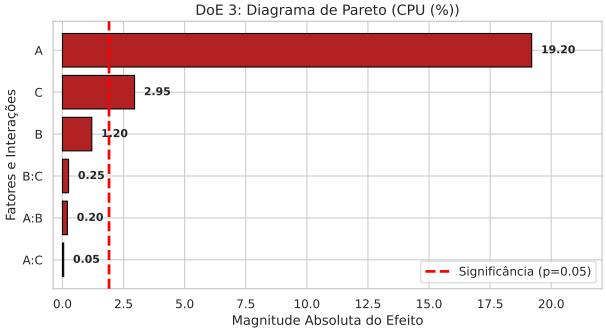

Figura 7: Diagrama de Pareto com o consumo gerado pelas configurações.

O Gráfico de Efeitos Principais (Figura 8) explica a natureza desse consumo. A reta do Fator A sobe bruscamente porque o sistema operacional Linux gasta energia para conseguir organizar os 4 processos trabalhando todos ao mesmo tempo. Já a linha do Fator C desce. Isso ocorre porque o pacote comprimido fica leve e transita rapidamente pela internet, evitando que o processador fique parado apenas aguardando a placa de rede terminar de transmitir os dados.

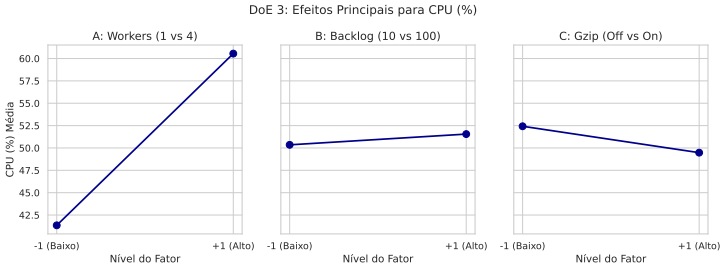

Figura 8: Como o gasto de CPU muda conforme a configuração.

O Gráfico de Interações (Figura 9) é muito importante para garantir que o servidor não vai travar. Como não existem cruzamentos nas retas, sabe-se que o aumento de gasto

de energia causado pelos *Workers* se mantém comportado e constante, mesmo quando o *Gzip* é ligado ou o *Backlog* é alterado.

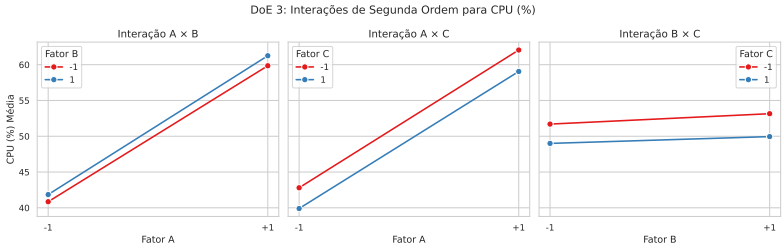

Figura 9: Interações comprovando o comportamento seguro do gasto de CPU.

7. Discussão e Comparação entre os DoEs

Olhando para os três testes (Latência, Throughput e CPU) de forma conjunta, consegue-se ter uma visão completa da saúde do servidor. Analisar apenas uma métrica isolada poderia levar a equipe a tomar decisões perigosas, como focar apenas na velocidade de resposta sem perceber o alto custo de processamento gerado. A comparação direta dos três cenários revela exatamente como as configurações do sistema interagem como um todo.

Fica claro que a alteração no Fator A (*Workers*) é a principal responsável por ditar o ritmo da máquina. Ao passar de 1 para 4 processos paralelos, observou-se a clássica troca de desempenho por consumo (o chamado *trade-off*). O servidor apresentou uma queda gigante no tempo de espera (redução de $159,00$ ms na latência) e um salto na quantidade de atendimentos (mais $20,98$ requisições por segundo). Em contrapartida, o preço dessa decisão foi o mais alto de todo o estudo, exigindo um esforço extra de $19,20\%$ do processador. Isso prova matematicamente que a grande âncora que segurava o servidor era a falta de processos trabalhando ao mesmo tempo.

A grande surpresa do estudo foi o comportamento do Fator C (*Gzip*), revelando-se a otimização mais inteligente e "barata"para a infraestrutura. Normalmente, o senso comum dita que usar ferramentas para compactar arquivos fará o processador trabalhar mais. Contudo, os testes provaram o oposto: além de ajudar a diminuir a latência ($-18,50$ ms) e aumentar levemente a vazão, o *Gzip* economizou $2,95\%$ de CPU. Isso acontece porque os dados ficam menores e viajam mais rápido pela rede. Com isso, o processador não fica tanto tempo ocioso apenas esperando a conexão de internet enviar a resposta, o que acaba poupando recursos gerais da máquina.

Por fim, o Fator B (*Backlog*) mostrou que não faz diferença matemática real em nenhuma das três áreas avaliadas. Isso indica que a fila de espera padrão de apenas 10 conexões já era mais do que suficiente para o tamanho do tráfego testado, e aumentar esse limite para 100 foi um ajuste que não trouxe benefícios. Somado a isso, o fato de não se encontrar nenhuma interação negativa (linhas cruzadas) nos três testes confirma que as configurações não brigam entre si. A equipe tem total segurança para aplicar estas melhorias em conjunto sem o risco de gerar instabilidades inesperadas no servidor.

8. Considerações Finais

Usar a técnica de Planejamento de Experimentos (DoE) 2^3 provou que não é preciso trabalhar na base da intuição e da "tentativa e erro"para configurar um servidor na vida real. A matemática e a estatística mostraram exatamente o peso de cada configuração.

A melhor configuração possível para este servidor web é usar o nível Alto de 4 *Workers* trabalhando de mãos dadas com a compressão *Gzip* ligada. Essa união deixa o servidor extremamente rápido, atendendo muita gente, com um gasto de processador que é facilmente justificado pelo ganho de velocidade. O tamanho da fila de *Backlog* pode ficar no padrão de fábrica, pois aumentar esse número não trouxe nenhuma melhoria.

8.1. Limitações e Trabalhos Futuros

Como limitação, é importante lembrar que este foi um teste controlado com tráfego previsível da ferramenta de testes. Na vida real, acessos que exigem muitas buscas pesadas no banco de dados ou download de arquivos gigantescos podem alterar esses resultados. Além disso, fez-se apenas uma rodada exata de matriz fatorial para tirar essas conclusões.

Para trabalhos futuros, sugere-se testar outras configurações de rede (como ajustar o limite de arquivos abertos do Linux ou ativar conexões *Keep-Alive*) e até mesmo testar servidores diferentes (como Nginx em vez de Gunicorn). Por fim, recomenda-se repetir a rodada de testes múltiplas vezes (*réplicas*) para se ter ainda mais certeza contra qualquer lentidão aleatória que a rede possa sofrer no dia a dia.

Referências

[1] MONTGOMERY, D. C. *Design and Analysis of Experiments*. 8. ed. Hoboken: John Wiley & Sons, 2012.

[2] JAIN, R. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. New York: Wiley-Interscience, 1991.

[3] SILVA, F. A. P. Projeto Final — DoE com dados de servidores web. Disciplina de Estatística, UFPI, 2026.

[4] CARVALHO, E.; FERREIRA, M. V.; LEAL, R.; SILVA, F. A. Análise de Desempenho de um Sistema Distribuído para Segmentação de Imagens da Retina. Universidade Federal do Piauí (UFPI), Picos, PI, Brasil.

[5] FRANCO, G. Entenda o que é doe – design of experiments. Escola EDTI, 2015. Disponível em: http://www.escolaedti.com.br/entenda-o-que-e-doe/